Prototype of Generic Server for Wind Power Plants Using IEC 61400-25 Standard

Olsen, Andreas Kargård; Ösdil, Baris; Poulsen, Bjarne; Pedersen, Knud Ole Helgesen; Johansen, Knud

Published in:
International Journal of Distributed Energy Resources

Publication date:
2007

PROTOTYPE OF GENERIC SERVER FOR WIND POWER PLANTS USING IEC 61400-25 STANDARD

Andreas Kargård Olsen, Baris Ösdil, Bjarne Poulsen, Knud Ole Helgesen Pedersen, Informatics and Mathematical Modelling, Centre for Electric Technology .
Technical University of Denmark Building 322, office 007, DK- Kongens Lyngby:Denmark
Phone (45)4525 5274, Fax +4545255300
E-mail:bjp@imm.dtu.dk & khp@oersted.dtu.dk
Knud Johansen. Q-Technology

Keywords: IEC 61400-25 Standard, Communication, Generic Server, Web Service, Wind Power Plant, Data Model, Service Oriented Architecture, Framework

ABSTRACT

The IEC 61400-25 international standard defines protocols for communication, control, and monitoring of wind power plants (WPP). The IEC61400-25 standard includes a wide range of mandatory and optional objects in the specified information and information exchange model, ranging from interfaces for control and monitoring to a standardized and secure way of handling communication for a WPP. An analysis focusing on isolating the necessary requirements has been carried out based on the IEC 61400-25 in order to create a generic prototype which can be used by WPP vendors. The main communications interface of the prototype utilizes web services and the prototype developed is comprised of several independent modules to allow for the possibility of choosing a fully customizable setup by the end user. Configuration of the system needs to be done in a simple way, ensuring a flexible and reusable system, where different choices for the system can be added or left out depending on user requirements. From the requirements a prototype with the purpose of examining the key aspects of these definitions has been elaborated.

1. INTRODUCTION

Wind power plants have through the years steadily gained a bigger and more dominant position in the power generation industry. Each vendor has their proprietary solutions on controlling and monitoring of the products supplied. In today's ever changing and rapidly growing energy market, monitoring of and easy communication between different systems, is essential. Through this communication the current state of the individual power plant can be controlled and monitored when required, and counter measurements can be enforced if needed in order to meet the changing demand for energy and keep the stability of the distribution system. It is vital that the overall dispatching systems are able to control the energy generation from a wind farm on demand in order to meet the fluctuations in the energy consumption. A common way to achieve this is a manufacturer independent approach.

As the complexity of the power distribution network increases, methods for efficient analysis, monitoring and coordination of the network control become essential. This in turn requires a highly efficient and dynamic control strategy for the power system network. Several organizations have addressed this issue in manners where the main objective has been to develop communication standards for interconnecting electric power generation systems.

One important effort towards standardization has been launched by the International Electrotechnical Commission, IEC. From the start, IEC focuses on communication within substations which has led to the IEC 61850 substation communication standard. Several standards for distributed power generation are under development using the information modelling concept, some of the object definitions and a part of the common data classes from the IEC 61850-7-2 and IEC61850-7-3 standard[1].

The vision of this project is to evaluate the new standard IEC 61400-25 for the WPPs and end up with a functional prototype implementation of the standard. In the future it is necessary that all new WPPs conform to the interface defined in the IEC 61400-25 architecture. This paper suggests how to implement a system for a WPP not providing an IEC 61400-25 compatible communication interface, but some of the ideas can also be used for one that does.

A framework encapsulating and compliant to the standard must be designed. This framework must expose the methods offered in the standard and taking care of all communication. The framework should be transparent so that a programmer can change the configuration of the framework. The approach for adding different layers to communication must be done on top of the communication model to ensure that different approaches can be used, for instance to secure communication, the framework will do most of the work involved in the transition automatically.

---

[1] If parts are not defined in the IEC 61400-25, they are often defined in the basic standard IEC 61850. In that case they have to be used.

The power distribution system is comprised of many different devices, distributed over a great physical space and requiring a variety of functionalities. SOA is a system architecture where a network is comprised of nodes exposing services to each other in order to complete a greater common task. The system will take advantage of this architecture where appropriate. Every WPP will be modeled as a web service taking care of the communication with different clients.

For the time being there has not been specified a common way to configure the system base on IEC 61400-25 standard or its constituting devices. In the future an extension to basic parts of standard will be released describing a configuration language. The prototype must be designed in a way where everything can be configured through a common interface.

The aspect of security is a major part of a system communicating data over the network. Security must be a configurationally option.

A massive amount of data is collected form the WPPs. These data has to be analyzed, filtered, and made accessible to clients as fast as possible. This leaves a lot of constraints on the storage device pertaining to access time, storage size and stability among others. In reflection of this, a great deal of work must be put into the design of fast access to the data and the storage structure making sure that it can meet the specifications that a system base on IEC 61400-25 standard demands.

2. ANALYSIS OF THE STANDARD AND THE REQUIREMENTS

The IEC 61400-25 series is a specialized version for defining and standardizing a unified communication for monitoring and control of wind power plants. One aim is to enable systems from different vendors to mutually communicate.

The IEC 61400-25 standard series is a wind power specific information modelling based on the modelling approach used in the IEC 61850 series of standards, which in general defines the communication aspects applied inside a substations. The IEC 61400-25 standard is domain specific use of the modelling concept in the IEC 61850 standard and reuses of object definitions and common data classed which in general could be apply to other power generation systems.

The standardization expands over the information modelling of the target system and the communication protocol for communicating the data encapsulated in the information model. As a result of this approach, the standard addresses the domain by separating it into three main categories of interest which together encompass all the important aspects of the communication and control of wind power plants. The three different main categories are as follows:

1. Wind power plants information models.(IEC 61400-25-2)
2. Information exchange models. (IEC 61400-25-3)
3. Mapping to communication profile. (IEC 61400-25-4)

2.1. Information Models – IEC 61400-25-2

The information model uses an Object Oriented approach for modelling the wind power plant components and data in general. It defines an exact model of the wind power plant (WPP) which contains the components and data of interest. This data will be made available to access for monitoring and control purposes. The information model constitutes a precisely defined set of data classes which will make it possible to build up a logical model of a specific WPP. The primary component in the model is a Logical Device (LD) which is defined as an abstract model able to fully represent a WPP. A LD residing on a server is assigned to a specific WPP which must be able to fully represent it with all its data and attributes. It must contain a collection of specific Logical Nodes (LN) which will further contain data instances reflecting the physical state of the WPP. Logical Device is a container for all WPP related data together with self descriptive meta-data describing the physical host and the device itself.

A logical node consists of a collection of related data, defined as Data Classes (DC). All the information in a logical node is contained in respective Data Classes. The structure of all logical nodes is similar and has a standardized form where different types of logical nodes can be constructed through the combination of different optional data classes.

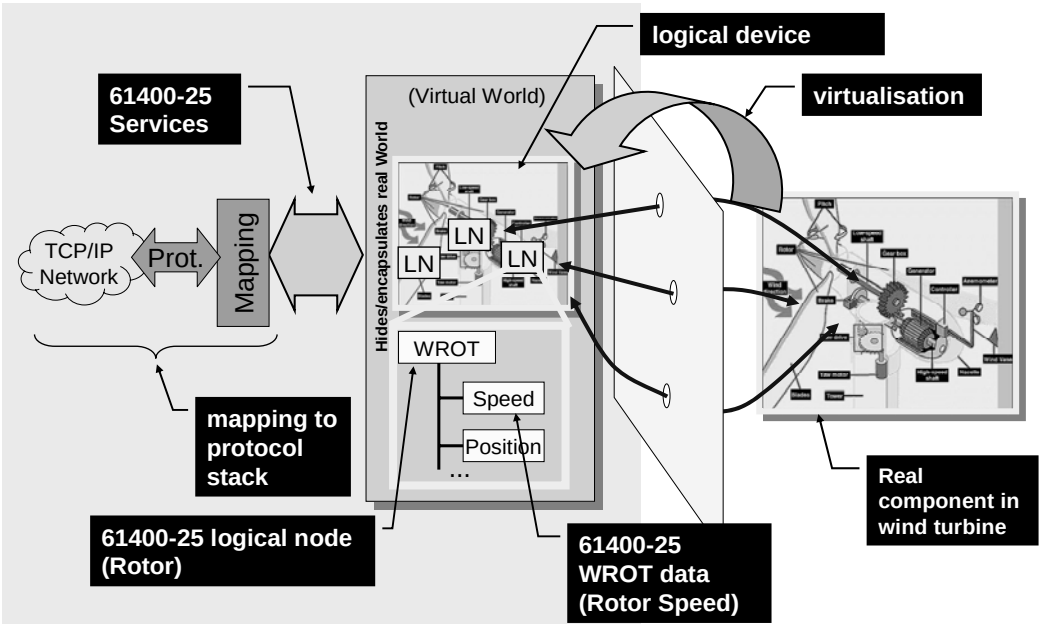

Figure 1. Modeling approach (IEC 61400-25-1)

All the logical nodes used in modelling the WPP inherit their structure from the abstract logical node class defined in IEC 61850-7-2. From an implementation point of view these different logical nodes will be similar since the structure is based on a common definition and follows a common pattern.

The hierarchical structure of a logical device residing on a server together with logical nodes and data classes is depicted in the figure 1 above.

The data class is the actual component of the information model which is used to define any data contained in logical nodes. These specific classes are called common data classes, which are precisely defined classes inherited from data classes. The common data classes used to model a wind power plant device can mainly be categorized under two groups. Common data classes a), defined specifically for wind power plants and b), logical nodes that are inherited from IEC 61850-7-3. A complete listing of these common data classes is provided in IEC 61400-25-2 clause 7.

The server must be capable of representing the information model with all its instances from logical devices and all the way down to specific data attributes. The hierarchical structure must be preserved so that each data instance can be referenced in a standard manner as defined in the information model.

A server may host one or more logical devices depending on the number of WPP which must be controlled by that server. Therefore it must be possible to uniquely reference a specific logical device representing a specific WPP.

The server must be capable of representing a logical node contained in a specific logical device and to refer to it uniquely. Each logical node has a unique reference name which makes it possible to locate it directly in conjunction with unique reference name of the logical device containing the logical node. This is important since a single server instance can have more than one logical device residing on it.

2.2. Information Exchange Models IEC 61400-25-3

This part of the standard describes the information exchange model which is implemented on the server enabling client's systems to access and modify data in the information model. Each information model instance has a service interface describing the operations available on that particular instance. Each information model object has a specific set of services making it possible to read or write from/to it.

The basic services that are used to mediate between the outside world and the real wind power plant device are referred to as Abstract Communication Service Interface (ACSI). The basics of these services are described in details in IEC 61850-7-1 and IEC 61850-7-2.

In the ACSI models, the information that gets reported or logged is represented by data sets. In that way reporting and logging can be defined in a more compact manner applying to a group of data.

The reporting services must make it possible to subscribe to spontaneous data reports on specific conditions for data values. Conditions such as change of value or change of attribute values will trigger a preconfigured reporting subscription and start dispatching the values.

In the reporting mechanism, the server must make the data for the client available for read and write. Since it is not common practice in client server architecture for a server to contact a client offering data, the server should buffer the values to deliver them later to the client, whenever a client request is made.

In order to achieve buffering mechanism so that the server does not have to notify the client for the available new data, some sort of a server side session must be implemented for the reporting interaction between client and server. The server must have an internal state making it possible to keep track of which step in the reporting process it is in, and which data has been sent and which is still in the buffer. However, the standard also specifies that it must be possible to configure the reporting mechanism such that data is not buffered in case of a connection interruption, meaning that the client only can access the data available at the time it makes its service request.

Like reports, logging can be initiated upon the client's request. The data can be logged on the same criteria, but in addition to this, updates also must be logged.

The client must be able to get a log at any time for a given interval. Due to the data storage amount it must be assumed that this interval is limited. Reports reflect current data while loggings reflect longer-term data, and system status.

2.3. Mapping to communication profile IEC 61400-25-4

The services defined in the information exchange model are mapped to standard web services. A detailed description of each service is provided together with the corresponding WSDL [1] describing the exact structure of the service methods. Each service defined for the various data models is mapped to SOAP services making it possible to transfer data with the correct types and structure defined in the information exchange model.

The server must resemble the web service description provided in the communication profile mapping. It should be possible to communicate between client and server accordingly as specified in the information exchange model.

3. CHALLENGES AND DESIGN REQUIREMENTS

The standards do not provide a solution for every single aspect of implementation and in many cases the standard leaves several choices or decisions for the system designer to solve. A presentation of the challenges in designing and implementing will be given next, together with a suitable solution strategy addressing the identified problems for the design of system based on IEC 61400-25 [2].

3.1. A framework based on components

The framework must take care of all the interaction with the protocol. Security, reliability, sessions et cetera. must be implemented in a standardized way. The framework must have a default setting for communication, but it should still be possible to change the different settings as needed. Looking at end-to-end communication, there are many single elements that must be easily changeable. For instance, each WPP might have its own way to supply the data. The system must be a very-late binding architecture, where one component's runtime is integrated with another component's runtime using dynamic invocation.

The system is to be designed using multiple modules that can be changed on demand. This implies that a common interface must be used. No matter how the mapping in the transport is done, the resulting requests and responses must be the same.

3.2. Configuration

The information model defined in IEC 61400-25-2 has a lot of optional nodes that can be implemented with optional data classes. The vendor must have a way to configure what their WPP has implemented, and how the communication is done. Providing such a configuration option will make reuse more efficient and it will be easy to set up a server representing specific WPPs. Configuration also includes information for how interaction between the WPP and the system must be performed, for instance how often can/must the data be pooled from the WPP.

The information model does not contain information of *how* the WPP and the system will communicate. It only states an abstract data format. Each vendor would have to design an entire system from information model to exchange model unless a common data input format is defined.

3.3. Data processing and data storage

To make a system that can handle the massive amount of data flow, it is vital that data can be processed in an efficient and secure way. The data must also be rapidly retrievable and easy to store.

On the server side, the information model must accurately reflect the hierarchical information structure. It is extremely difficult to represent the model using a classic relational database. Besides the difficulty in representing the model, data retrieval will also be extremely slow. This is because the server must execute a large number of queries to retrieve data from the complex relations in the data hierarchy. Instead of storing the current data on a persistent storage for service retrievals, the information model could be constructed on the server process itself making use of the representational power of an object-oriented language. The server process could programmatically replicate the information model and store the data in its run-time with suitable objects while running and servicing the client systems. This approach will make the execution very fast since the data is already stored in main

memory ready to be fetched. And the modelling/representation of the information will be straightforward through instantiating objects from class definitions resembling the class definitions in the information model.

To store the most current data and make it available for client processes, the server must continuously pool the WPP for new data made available. Only the latest data is kept inside the system to speed up data handling. Long-term data is separated from the running system to cut down runtime challenges. The information model is modelled in an object-oriented way, to keep the implementation and the information model close together.

3.4. Security

IEC 61400-25-3 defines the security aspects for the standard and how to solve it in general, but how it is handled specifically is completely up to the individual supplier.

One supplier might simply use a secure line, and therefore remove any security aspects of the service itself, while another might want to use a public ISP where the service related traffic must be secured by the service itself. This calls for a solution where security is built on top of the communication as a separate layer in a modular fashion easy to add, remove or change on demand

In TC 57 [3] a proposal has been presented in which a security model has been suggested.

In this research TCP/IP traffic is utilized, and therefore has to follow the IEC 62351-3 security standard. The standard recommends transport layer security (TLS) to tackle the most common security threats. At the same time it specifies that security must follow progress and update to better solutions when available.

Another aspect of security is access control. Access control has the duty of ensuring that only authorized individuals can gain access to the data. How and what the security includes is defined by the service mappings (SCSM).

In IEC 61400-25 the minimum requirement for access control is only defined as the need for supplying a valid username and password to gain access. This ensures that only people with a valid password can gain access to the system. Different users might have the rights to perform different actions.

The system only contains methods and data the client can access. The user either has the right to run a method or not. Through the method the client can either read or write data by functional constraints. Every client does not necessarily have the right to view all data but maybe only a subset.

Users can be granted read or write rights to specific nodes. In the same way users can get clearance to invoke specific methods. Each node or method must have lists of what each user is allowed to do.

4. PROPOSALS FOR A GENERIC ARCHITECTURE

To make the system generic, different elements in the system must be isolated. The figure below shows a proposal for the system architecture.

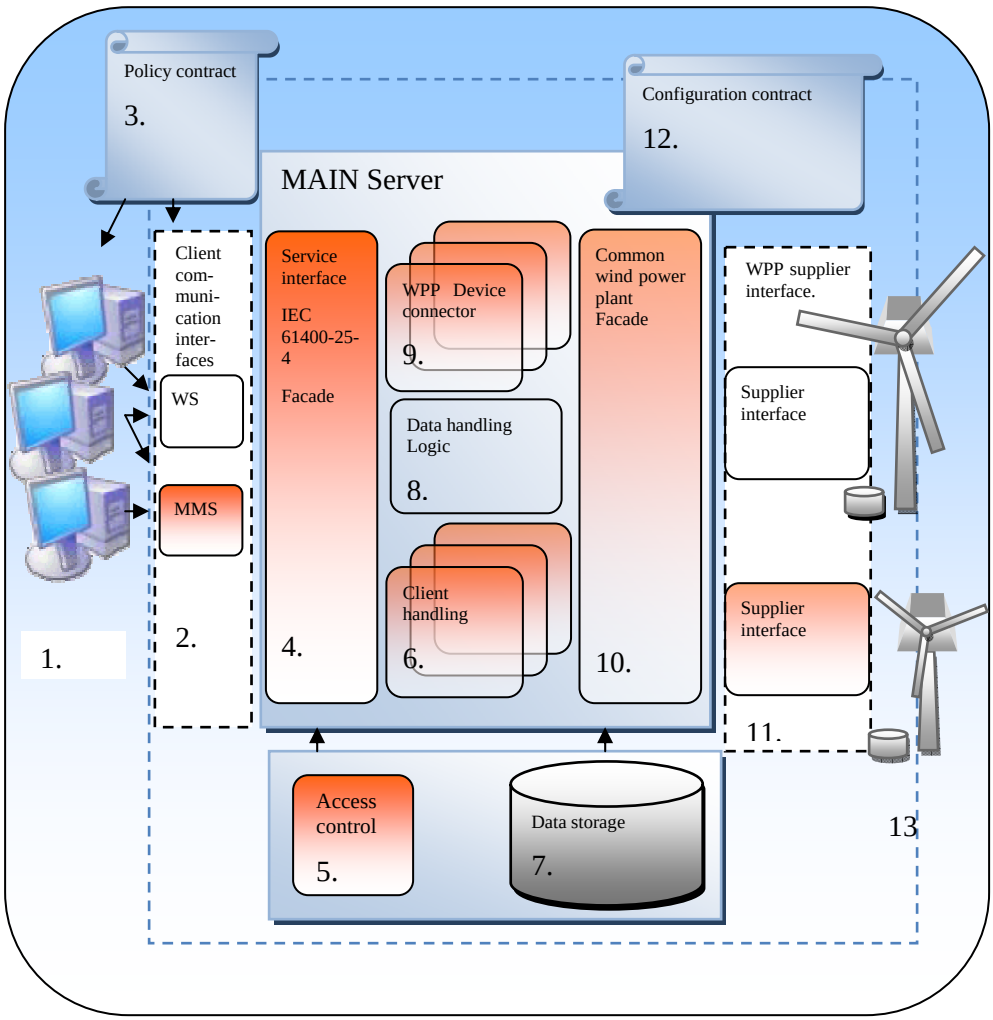

Figure 2. Proposal for a system architecture.

1. Clients can be web clients, applications et cetera. (The client is outside the scope, and should simply be an end user of the service.)

2. The client can use different methods to conduct the communication with the service. The main focus is on web services but it can equally be an MMS or others. The interface is defined in IEC 61400-25-4.

3. Policy files define specific information for the communication. This includes security information, use of transport protocol, et cetera. Policy files are ideal for custom configuration at deploy time.

4. A common interface assures that the service can always be reached through the client communication interface as long as the client obligates himself to use the interface defined in IEC 61400-25-4.

5. The access control checks the client's right to access a specific method or data-set.

6. When communicating with the service the client can ask for specific data. The client handler manages the connection between different clients and the service. The information exchange model defines how this is done in IEC 61400-25-3.

7. Stores the data for shorter or longer time.

8. Data handler logic keeps track of data, and the logic to run the service generally.

9. Different WPP suppliers produce different data. The pooler gathers the necessary data from each WPP. The information model in IEC 61400-25-2 defines this.

10. There is no generally accepted standard for how data must be supplied by the WPP, but by defining one, supplier modules are easier to replace in the system.

11. Supplier interface. When connecting a WPP to the service, the supplier has to create an interface file. The file has knowledge of the supplied nodes that the WPP can expose, and how often pooling is necessary.

12. The contract must contain information of the data each WPP can deliver. Also the contract defines how often the data is to be pooled. This must be done in compliance to the IEC 61400-25-2 information model.

13. Each supplier exposes its capabilities to get and set data through their own interface.

5. CASE STUDIES

The purpose of these case studies is to try some of the different scenarios that the system is supposed to work under. The overall design of the system has been built in order to have the same basic design for all the challenges, and then make minor changes in order to add or remove functionalities to the system. Before studying each case, the setup, overall architecture and the internal workings of the server system are described. The prototype focuses on web services.

5.1. System Description

The core unit of the system is a server process hidden behind the service interface where device data and client requests are handled. It has two important interfaces

through which it accomplishes its tasks. These are, respectively, first the interface to the communication service which acts as a bridge between client processes and the server, and second, an interface which can be configured to pool data from WWP. The data retrieved from the configured device(s) is kept in an internal, in-memory data model for fast retrieval and processing when client requests are being handled.

IEC 61400-25 defines an event driven service model for monitoring WWP. Especially in the reporting process various conditions have to be monitored continuously as new data arrives, so that proper actions can be taken in order to satisfy each report configuration which will serve the client process with information.

Service Interface - Communication Module

The Service is supposed to work with different client protocols. One of these is Web services, but the system should also be able to work with other protocols like MMS. It is possible to implement the protocol directly in the server, but a more flexible solution is to have one server to serve the different communication modules through a well known protocol. A facade presenting an interface can be created. Each protocol then has to implement a module that takes care of getting information from the client and returning the server reply to the client. This conforms to the data defined in IEC 61400-25-3. This approach enables a more modular architecture for the system.

Device Interface

The communication interface to WWP from the main server is beyond the scope of the IEC 61400-25 series. No specification is given on how to handle the communication with devices that are supposed to be monitored and controlled. However, in order to be able to test the server prototype, a simulator is implemented and connected to the system. The simulator acts as a device continuously generating data for a specified set of data objects, for a pre-specified logical device data model defined on the server. The data is extracted from log files which contains real-time stamped wind power plant device data. The data entries contained in the log files are mapped to data types defined in IEC 61400-25 where suitable.

Server Configuration

Initially at start up, the server must be configured in order to work properly and interact with devices and communication services attached to it. At start up it will configure itself through an XML configuration file. The information model reflecting device data, device connections, pre-configured data sets and access control configurations are all made through dedicated XML files for each specific configuration.

Configuring the information model

At start up the server will first create the data structures reflecting each specific device attached to it in order to be able to house the data and make it available to client systems. The XML configuration file makes it possible for a server administrator to create data structures for logical devices with its logical nodes, data ob-

jects and data attributes. Initially the configuration file defining the data structure with their default values is parsed, where afterwards the object instances are created on the server according to the structure defined in the configuration file. It is possible to initiate data structures for several logical devices in the same configuration file.

The configuration file resides with the server. The server must know the exact data structure of the physical devices which are to be attached to the server before composing the configuration file. In the future the configuration could be done dynamically by retrieving configuration information directly from the physical device itself.

5.2. Case study 1

This case study has been focused on the following subjects:

- Development of a generic device communicator for acquiring data from different WPP data suppliers
- Securing access and data across a web service
- Cross-language access, and different clients

This case has shown that web service clients in both Java and WCF can be connected to the web service, and retrieve the information gathered for different WPPs. It is also shown that a web client can gain access to the service through the use of a web server.

The device connectors could act by it self or represent several WPPs. The gathered data could be collected and managed in one central server.

Security can be addressed in a seamless and uncomplicated fashion following security patterns.

5.3. Case study 2

In this case study, the focus is on a virtual major corporation owning a big wind farm. They operate many WPPs and monitor the system from one or more locations. Some of the WPPs are connected using a modem. For this reason it is important to get the most out of the bandwidth by compressing the data transmitted. In this scenario, the communication line between the WPPs and the server can be lost. In case of connection loss it must be possible to recover the data produced during the outage.

Some of the WPPs have their own device connectors and others have a joint connector. Within the corporation many people must be able to view the data simultaneously. The corporation has a dedicated secure private line for internal communication. No security is needed concerning this. However, data must be protected internally. Not everyone in the corporation is allowed to access data, and even if

access is allowed, not everyone is allowed to run all service methods on the WPP devices. Access control via a replicable module is included.

Key points:

- Many WPPs are hosted under one server.
- Multiple clients.
- Avoiding data loss during offline period.
- Compression for lower bandwidth usage.
- Role based access control.

Offline scenario

If connection is lost, data must be stored until the client can retrieve it. An extra or redundant server connection could provide an alternative way to upload data, but since the WPP is supposed to be autonomous and recover from critical situation itself, it must be concluded that information is not time critical at a level, and this solution is routed out, in order to not have the extra cost of maintaining several communication lines. A much cheaper method is to use hardware already placed at the WPP. In such a case each device connector keeps a buffer of data which can be retrieved on demand. However, if the data is not retrieved by the client, the list will keep on growing in memory and eventually consume all of the memory resources.

To overcome this problem data must be swapped to disk on regular intervals. This will not only reduce the memory usage, but it will also assure that data is not lost if the machine hosting the data connector crashes.

The device connector works with time stamped packages. If the buffer holds more than a given number of entries, the data will be saved to disk, and the stack will be emptied. On the next attempt by the server to retrieve data, the retrieved object will contain a flag telling the server that buffered data exists. The server can retrieve this data on demand. Both size of the stack, file size and transferral web service transferral buffer must be configurable.

If the connection is poor, but do exist, reliable sessions are a great addition to the quality of the communication. “Reliable sessions” or “reliable messaging” assures that a message will be re-sent if communication is disrupted. It will keep on trying until the message has been delivered, but at the same time uphold an assurance of exactly-once and in-order delivery of the messages. Reliable messages are part of several of the standard bindings, and can be turned on and off by demand. Like other attributes, this can be done either in code or in the configuration, by enabling reliable sessions. Reliable messaging is turned on for the prototype.

Access control (AC)

To ensure that only the right people have access to data, an access control system is needed. The standard proposes an access control system, but it has not yet been

fully specified. For the purpose of the prototype, a system for controlling the access to the system has been implemented. This is built as a role base access control system. (RBAC).

Ensuring module exchangeability

The modules in the system should be interchangeable. Modules are accessed from many different methods in the server, and as such they are incorporated deeply in the system. The solution lies in linking the object during start-up rather than under compilation. The principal is described by Martin Fowler as a plug-in pattern [4]. This pattern is one of the corner stones in order to have interchangeable dynamic modules. A negative effect is that DLL access is slower than if the linking was done at compile time. But this is a minor expense compared to the big flexibility it provides for the system. The downside to this approach is that all modules must be conforming to the same module interface, and therefore the interface must be finalized.

Minimizing traffic

SOAP is not a good format when considering the amount of traffic generated. It is in plain-text format and contains lots of meta-data to describe the message. This makes SOAP a perfect candidate for compression.

For web services using WCF there are three main ways to encode the data for transport. These are text, binary and MTOM [5]. Text is plain SOAP in ASCII encoding; this of course is the most compatible way to send the data, but also the most expensive with respect to bandwidth consumption. MTOM is a WS standard, and can compress the body of the message. MTOM can also be used for transporting binary data between services. This comes in handy if files should be transported from a server to a client, but since it still has to be transformed to base64 encoded data, and still uses SOAP wrapping, alternative methods like FTP transport would probably be a better solution for this. Different bindings or compression do reduce the traffic drastically, but it does slow down the system because of the extra computation needed [6,2]

Binary traffic is the most optimal way to transport the data, but is restricted to a windows-to-windows communication. However, even if the communication is restricted to windows-to-windows communication, this is the best choice.

The last approach for reducing traffic is to create a custom encoding schema. For instance, the traffic could be encoded using one of the commonly used compression libraries, like ZIP or RAR. This of course will demand that both the client and the service are able to encode and decode the traffic. Compression is specified in the HTTP 1.1 protocol.

Simulating the wind farm

Each device connector must be initialised at start-up. The server knows where and how and when data must be retrieved, and will in turn collect it.

The clients contact the service through the Internet information server. This server assures that multiple clients can interact with the service at the same time.

Case summary

A device connector module is able to store data for later retrieval from the server. The module provides a way to get data whenever possible, in manageable data packages. This ensures data delivery, and at the same time reduces peek loads on both the device connector module and the server. No standard is defined for how the data is conveyed from the physical device to the server, but the connector (device simulator) did provide some basic cases that all connector should address no matter how they are designed internally. The lack of a standard for a WPP communication makes it hard to reuse the system between different vendors, because the module will be incorporated into the server, unless a common interface is defined. However, with a defined interface at hand, the system would be very flexible and have opportunities for changing the modules as needed.

5.4. Case study 3

Case study 3 looks at some of the different bindings and protocols that can be used in the communication.

The main aspects of the case study are:

- Tests with different encoding, different security choices and compression.
- Relative speed.
- Relative number of calls serviced.
- Multiple end points for better flexibility and performance for the clients.

It is worth remembering that most of the bindings support the same basic features like security, reliable messages, different encoding schemes, and as such the basic features can be kept no matter which binding is chosen.

A test shows that the different standard encodings did influence both speed and data amount sent. The test has shown that the system did react to different configuration parameters.

Due to the nature of web services and the way they are configured by using for instance Windows Communication Foundation (WCF) [7,8], it is easy and cheap to provide several ways for clients to communicate with the system.

Additional connection policies can be added simply by supplying an endpoint to the configuration file. Not all clients support all of the possible choices the service can communicate. By exposing multiple endpoints each client can choose that endpoint providing the highest efficiency for him. This ensures that the system does not have to operate only under conditions designed for a worst case scenario, but it

can provide different levels of service. This maximises flexibility for both client and service providers.

5.5. Case study 4

Part 5 of the IEC 61400-25 series is dedicated to conformance testing. The completed system should be tested successfully against all the proposed tests. What to test for varies a great deal but the most common attributes a system should have includes capability, reliability, efficiency, portability, maintainability, compatibility and usability.

The tests conducted for the developed prototype is limited to the functional testing of the services offered and general unit testing during development of the prototype. The aim of functional testing was to test whether the service methods operated as they should with proper request and reply objects.

The service tests conducted shows that the implemented service methods can generate the expected responses with the correct data types defined for the services. This implies that the business logic executed on the server is correct as well.

6. CONCLUSION

The goal was to implement and test an IEC 61400-25 compliant generic server which can be used to monitor and control wind power plants. During the development of the server there has been used real log data from wind power plants.

The main focus has been on the server system and its interaction with clients through the specified web services defined in IEC 61400-25-4. When developing the server system, general considerations such as security, modularized software architecture has been taken into account and implemented, whenever applicable.

Almost all services defined in the standard have been implemented using web services. This includes mapping and representation of data on the server side and to communicate the data to client systems.

The IEC 61400-25 standardization series is still “work in progress”, and during this research several updates have been released. The new releases of the standard were mainly dealt with ACSI service mappings to a communication profile. This is part-4 of the series defining the web service interface with its data types. At a certain stage the project had to be closed for changes at some stage in order to finish the implementation. However, services implemented lately are in conformance with the later versions of the mappings.

The list below outlines the main requirements which have been implemented in this project.

Information Model
On the main server, a suitable data structure has been set up which can store data as defined in IEC 61400-25-2. It is an in-memory data structure making data manipu-

lation and retrieval very fast and efficient. When building logical devices from logical nodes and data instances, the exact hierarchical structure can be presented on the server. The information model on the server is initialized through an XML file which makes it possible to alter the data models of the different devices easily.

Service interface
The web service interface, through which the service methods are communicated, has been implemented. The web service is implemented from the WSDL given in IEC 61400-25-4. WCF has been used for the web service in order to benefit from latest improvements within web services.

Different end points providing different communication features can be set up for the service so that client systems can choose the best suitable endpoint. Tests in case study 3 shows that different choices regarding communication (encoding, encryption, and transport protocol) will result in big performance variations.

Services
The server logic for the services has been implemented. Business logic within the different modules has been isolated from the server. It is possible to send service request with the correct data types defined in the standard and retrieve the corresponding response.

Module based architecture
The server components have been implemented as separate modules which make it easy to configure and change the system. For example the service interface is not tightly coupled with the central server itself.

It is possible to attach another service interface using another protocol than SOAP/web services.

The change of the AC module is only a matter of replacing a DLL in a directory. It is not even necessary to provide any configuration, changes, the system will automatically recognize that a module containing AC information is present, and it will automatically load it into the running system.

Reporting
The reporting service has been implemented according to the event driven data retrieval model defined in the standard. A client system can subscribe for reports which are generated by the report control blocks initialized by the subscription service. Both buffered and un-buffered reporting has been implemented.

Access Control
An access control mechanism is implemented making it possible to create a specific view of the data and services applied to it. The access control module can be configured so that different views of the system can be assigned to different client systems.

Device simulators/connectors
In order to be able to test the system, device simulators capable of generating data have been implemented. The simulators uses log files from real wind power plants when generating data. Different scenarios, which are most likely to appear in a real world case, have been studied, such as connection loss and lost data packages.

Configuration
The server and its modules are implemented such that they can be configured easily. For example the access control mechanism can be configured through XML files in order to create different views of the system.

The server can be configured to handle different WPP simulators with different data models. The structure of each simulator attached to the system can be defined separately.

The device connectors can be configured so that the server can locate the simulators and associate it with the respective data model defined on the server.
The security and transport features of the system can easily be configured through configuration files.

Clients
A web based client has been developed to test and demonstrate the services implemented. The implemented communication module for the server is based on web service. Any client capable of consuming web services can connect to the server and perform its tasks through the web services made available.

Overall Conclusion

The main focus was on creating a system to implement a server system compliant with IEC 61400-25 standardization series, a system that was not tightly bound to specific technologies and easy to configure. The resulting implemented server system has shown that such a system is possible.

The system covers most of the major parts defined in the standard. All topics in the requirements has been addressed at some level, and we believe this research should provide a good basis for future work and efforts toward the great goal of creating a vendor independent communication environment for all wind power plants in the world.

6. ACKNOWLEDGEMENT

We gratefully acknowledges the support from Claus Bjerge E2 part of DONG energy for real data from wind farms and the contributing with knowledge about Wind Power Plants. We would also like to acknowledges Aksel Kargaard Olsen for reviewing and commenting on this document.

7. REFERENCES

[1]Web service description language http://www.w3.org/TR/wsdl20/

[2]Andreas K. Olsen, Baris Özdil Prototype for a IEC 61400-25 Compliant Generic Server DTU 2006

[3] Frances Cleveland IEC TC57 Security Standards for the Power System's Information Infrastructure – Beyond Simple Encryption

[4] Martin fowler Patterns of Enterprise Application Architecture Addison-Wesley 2002

[5]SOAP Message Transmission Optimization Mechanism http://www.w3.org/TR/soap12-mtom/

[6]Christopher Kohlhoff Evaluationg SOAP for High Performance Business Applications: Real-Time Trading Systems.

[7] Keith Brown security Brifes http://msdn.microsoft.com/msdnmag/issues/06/08/SecurityBriefs/default.aspx#S3

[8]David Pollmann Programming INDIGO Microsoft press 2006